National Research Council Canada
Conseil national de recherches Canada

Division of Electrical Engineering
Division de génie électrique

ERB-974

Efficient Solution for the Kinematic Velocities and Accelerations for the PUMA 560 Robot

S. Elgazzar

October 1985

Canada

NRCC No. 25052

86-02690/

Copyright 1985 by
National Research Council of Canada

Permission is granted to quote short excerpts and to reproduce figures and tables from this report, provided that the source of such material is fully acknowledged.

Copyright 1985 par
Conseil national de recherches du Canada

Il est permis de citer de courts extraits et de reproduire des figures ou tableaux du présent rapport, à condition d'en identifier clairement la source.

Additional copies are available free of charge from:

Editorial Office, Room 301
Division of Electrical Engineering
National Research Council of Canada
Ottawa, Ontario, Canada
K1A 0R8

Des exemplaires supplémentaires gratuits peuvent être obtenus à l'adresse suivante:

Bureau des publications, Pièce 301
Division de génie électrique
Conseil national de recherches du Canada
Ottawa (Ontario) Canada
K1A 0R8

EFFICIENT SOLUTION FOR THE KINEMATIC VELOCITIES AND ACCELERATIONS FOR THE PUMA 560 ROBOT

S. Elgazzar

ERB-974
NRCC No. 25052

OCTOBER 1985

TABLE OF CONTENTS

		Page
ABSTRACT/RÉSUMÉ		v
CHAPTER 1.	INTRODUCTION	1
CHAPTER 2.	DEFINITIONS	2
2.1	Robot Joint Variables and Cartesian Vector	2
2.2	Arm Configuration	6
2.3	Singular Points	6
CHAPTER 3.	VELOCITY TRANSFORMATION	7
3.1	Cartesian-to-joint-space Velocity Transformation	8
3.2	Joint-space-to-Cartesian Velocity Transformation	12
CHAPTER 4.	ACCELERATION TRANSFORMATION	13
4.1	Cartesian-to-joint-space Acceleration Transformation	13
4.2	Joint-space-to-Cartesian Transformation	18
CHAPTER 5.	SUMMARY AND DISCUSSION	19
REFERENCES		20
APPENDIX A.	SOLUTION FOR KINEMATIC POSITIONS	21
A.1	Direct Kinematic Solution	21
A.2	Inverse Kinematic Solution	21
APPENDIX B.	PROGRAM LISTINGS	23

LIST OF FIGURES

1	Robot arm: degrees of joint rotation and member identification	3
2	(a) Robot arm: initial state (b) Representation of link ℓ_3	4
3	Link representation for the PUMA 560 Arm	5
4	Coordinate systems for the first three joints of the PUMA 560 arm	8

ABSTRACT

An efficient solution for the kinematic velocities and accelerations for the six-degree-of-freedom PUMA 560 robot is presented. Kinematic velocities and accelerations are defined as the transformation of the linear and angular velocities and accelerations of the end effector to joint velocities and accelerations and vice-versa. The solution presented in this report builds on the results of the kinematic positions in Elgazzar ("Efficient solution for the kinematic positions for PUMA 560 robot." NRC/ERB-973, December 1984.) The method used in all kinematic transformations is based on the Featherstone method where the special geometry of the robot is fully exploited in the derivation of the solution.

RÉSUMÉ

Une solution efficace pour la détermination des vitesses et des accélérations cinématiques du robot PUMA 560 à 6 degrés de liberté est présentée. Les vitesses et accélérations cinématiques sont définies comme la transformation des vitesses et accélérations linéaires et angulaires de l'effecteur final en vitesses et accélérations au niveau des articulations et vice versa. La solution présentée dans le rapport est construite à partir des résultats des positions cinématiques d'Elgazzar ("Efficient solution for the kinematic positions for PUMA 560 robot." CNRC/ERB-973, décembre 1984.) La méthode utilisée dans toutes les transformations cinématiques est basée sur la méthode Featherstone qui exploite à fond la géométrie particulière du robot dans le calcul de la solution.

EFFICIENT SOLUTION FOR THE KINEMATIC VELOCITIES AND ACCELERATIONS FOR THE PUMA 560 ROBOT

S. Elgazzar

CHAPTER 1. INTRODUCTION

We consider in this report the solutions for the kinematic velocities and accelerations of the PUMA 560 robot, a computer-controlled manipulator consisting of six degrees of freedom.

The kinematic problem deals with the analytical study of the spatial configuration of the robot arm as a function of time, in particular the relation between the joint variable space and the Cartesian space. The transformation between the two spaces is very important since robots are usually servoed in the joint space whereas tasks are defined and objects manipulated in the Cartesian space.

The kinematic problem consists of two sub-problems: the direct kinematic problem, which is defined as the transformation from the joint space to the Cartesian space, and the inverse kinematic problem, which is defined as the transformation from the Cartesian space to the joint space.

To simplify the solution of the problem, the manipulators are usually designed so that neighbouring joint axes are orthogonal to each other and the last three joints have their axes intersecting at a point (i.e., spherical wrist). The spherical wrist allows the decomposition of the six-degree-of-freedom kinematic problem into two three-degree-of-freedom kinematic sub-problems representing position and orientation.

Reasonably efficient algorithms have been developed for these transformations. The Featherstone method (1, 2) seems to be the most efficient method (particular to a specific robot), for solving for the kinematic positions and velocities. Hollerbach (3) extended this technique to calculate the inverse kinematic accelerations. The algorithms developed are specific to manipulators having co-planar links with six revolute joints and a spherical wrist. The PUMA robot is different from a co-planar robot by having offsets at the shoulder and elbow joints and also having an asymmetry in the forearm link. These differences affect the solution of the first three joints.

Elgazzar (4) presented an efficient closed-form solution to perform direct and inverse kinematic positions for the PUMA 560 robot. Special attention was given to the arm configuration, and a method to obtain the configuration parameters from a known set of joint angles was presented.

The robot definitions, chapter 2 in ref. 4, are included in this report for completeness. The equations representing the solution for the kinematic positions are included in Appendix A for reference. It is assumed that the reader is aware of their derivation and significance; if not, it is suggested that ref. 4 be reviewed.

In this report, we present an efficient algorithm for the calculation of the kinematic velocities and accelerations for the PUMA 560 robot. In order to find the kinematic velocities (accelerations), one uses intermediate and final results of the kinematic positions (velocities). Thus, a significant amount of duplicate calculation on the part of the velocity and acceleration transformations is avoided, resulting in a substantial increase in speed over the general method of inverse transformations. The algorithm is applicable to the PUMA 560 robot and relies on the same features of geometry as does the position transformation ref. 4. The listings of the C-programs used in this study are given in Appendix B.

CHAPTER 2. DEFINITIONS

(Chapter 2 is reprinted from ref. 4)

In this chapter a description of the PUMA 560 robot is presented. The reference state of the arm and the arm configuration parameters are defined. A brief discussion of the singularities is also given.

2.1 Robot Joint Variables and Cartesian Vector

The PUMA 560 robot has six revolute joints arranged as shown in Fig. 1.

Joint 1 axis coincides with the centerline of the "trunk", link ℓ_1, which coincides with the z-axis of the base coordinate system. The joint angle θ_1 is measured in the counterclockwise direction from the positive y-axis.

The joint 2 axis is perpendicular to and intersects the joint 1 axis and coincides with the centerline of the "shoulder". The "shoulder" is the offset of length d_1 between the "trunk" and the "upper arm". This offset is parallel to the x-y plane and is in the negative x-axis direction when θ_1 is equal to zero. Link ℓ_2, the "upper arm", rotates around joint 2 axis an angle θ_2. The angle θ_2 is equal to zero when link ℓ_2 is parallel to the z-axis, as shown in Fig. 2a.

Joint 3, the "elbow", has its axis parallel to the joint 2 axis. The "forearm", the third link, is formed of a two-part link labelled a and b in Fig. 2b. Link ℓ_3 is defined as the vector sum of a and b, that is, the distance between the joint 3 axis and the center of the spherical wrist.

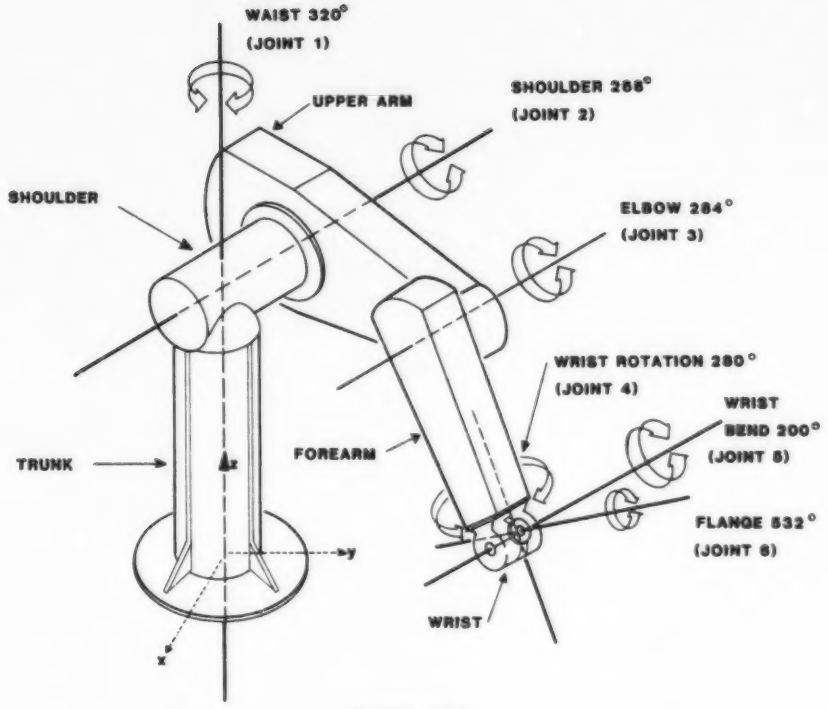


Figure 1. Robot arm: degrees of joint rotation and member identification (reprinted from ref. 4)

Link ℓ_3 is offset from link ℓ_2 by a distance d_2 parallel to the x-y plane and rotates an angle θ_3 with respect to link ℓ_2. When θ_1 is equal to zero, d_2 is in or parallel to the positive x-direction. At the reference state, that is when the arm is pointing up (Fig. 2a), link ℓ_2 is parallel to the z-axis and link ℓ_3 makes a known angle δ with the vertical. This angle δ is a function of the arm dimensions. Thus, at the reference position

[2.1] $\theta_{3r} = -\delta$

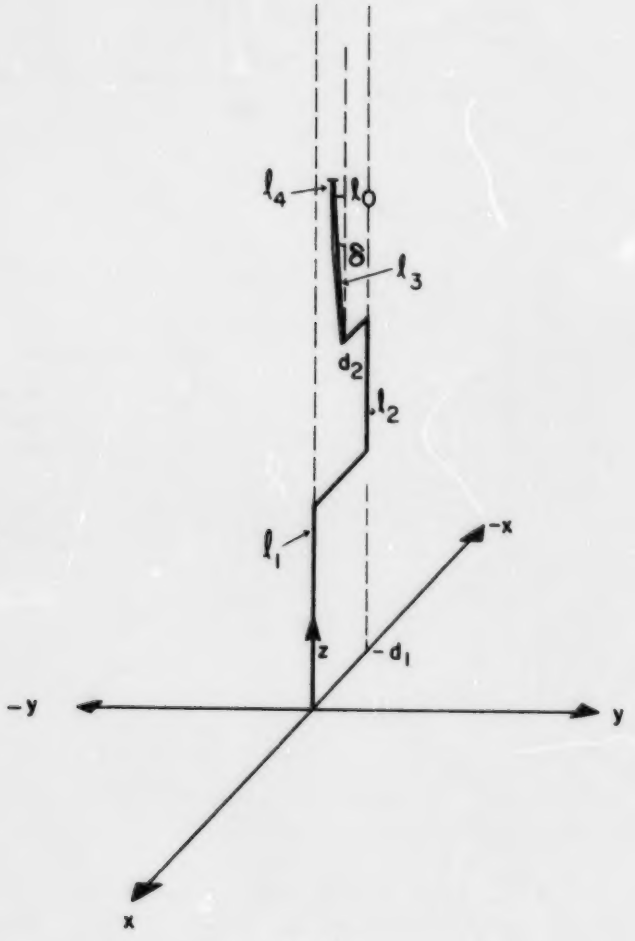
where

[2.2] $\delta = \sin^{-1}(\ell_o/\ell_3)$

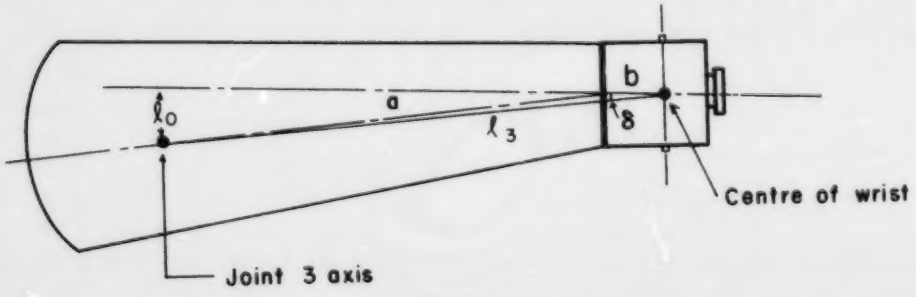
"ℓ_o" is defined as the offset between the centerline of the two parts forming the "forearm", link ℓ_3.

Joints 4, 5, and 6 form a spherical wrist.

Joint 4 axis is perpendicular to and intersects joint 5 axis. Link



(a)



(b)

Figure 2. (a) Robot arm: initial state (b) Representation of link ℓ_3 (reprinted from ref. 4)

ℓ_4 is the link from the center of the wrist to the flange or to the end effector if connected. Joint 4 rotates an angle θ_4 around its axis.

The joint 5 axis is parallel to the joints 2 and 3 axes. The angle θ_5 is the angle of rotation of link ℓ_4 measured with respect to the z-axis in the coordinates of link 4 (i.e., rotating the base coordinates through $\theta_1 \underline{k}$, then through $-(\theta_2 + \theta_3) \underline{i}$ and finally through $\theta_4 \underline{k}$).

The joint 6 axis is perpendicular to and intersects the joint 5 axis. It coincides with the centerline of the gripper mounting flange.

The intersection of the latter three axes in a point is shown in Fig. 1.

The link representation of the robot arm at any arbitrary position is given in Fig. 3. The position of the flange (or alternately the end effector or tool) is expressed in joint coordinates as

[2.3] $\underline{\theta} = (\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6)$

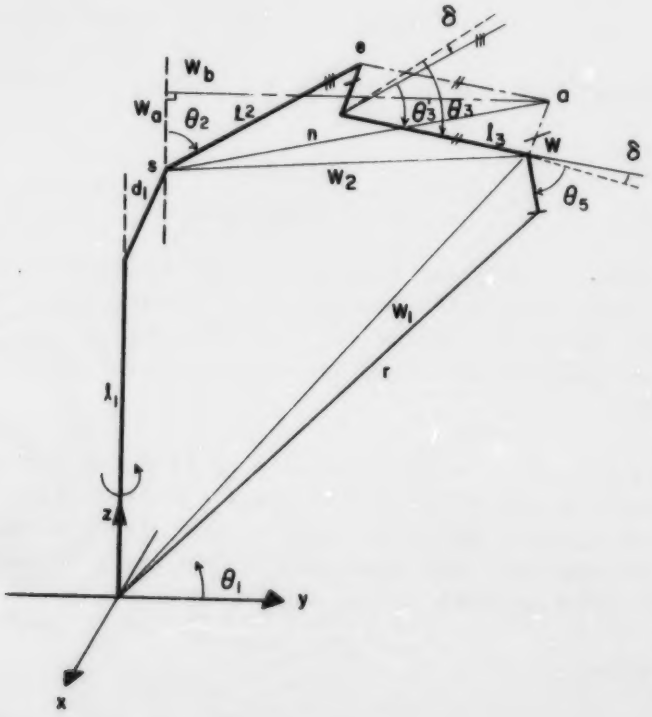


Figure 3. Link representation for the PUMA 560 Arm (reprinted from ref. 4)

and in Cartesian coordinates as

[2.4] $\underline{R} = (r_x, r_y, r_z, r_\rho, r_\theta, r_\psi)$

where $\underline{r} = (r_x, r_y, r_z)$ is the position vector and r_ρ, r_θ, r_ψ are the rotations about the z axis, the new -x axis, and the new z axis that aligns the base coordinates with the tip coordinates. The rotation parameters were chosen because they correspond with the arrangement of the joints at the wrist and hence simplify the solution.

2.2 Arm Configuration

The robot arm has similarities with the human arm geometry. Robot manipulators are defined accordingly as having a shoulder, an elbow, and a wrist; see Fig. 1. The robot arm may be righty or lefty, that is, the first three joints of the robot resemble a human's right or left arm, respectively. The elbow can have two configurations; elbow up, where the elbow's position is above the line joining the shoulder to the wrist; and elbow down, where the elbow's position is below that line. Also, the wrist can assume two solutions: the no-flip wrist for which the joint 5 angle is positive and the flip wrist for which the joint 5 angle is negative.

The arm configuration parameters k_1, k_2, and k_3 are defined as follows:

$$k_1 = \begin{matrix} +1 & \text{lefty} \\ -1 & \text{righty} \end{matrix}, \quad k_2 = \begin{matrix} +1 & \text{elbow up} \\ -1 & \text{elbow down} \end{matrix}, \quad k_3 = \begin{matrix} +1 & \text{no-flip} \\ -1 & \text{flip} \end{matrix}$$

From the above, it is clear that there are, in general, eight sets of joint angles for each attainable position and orientation. In practice, however, restrictions on the ranges of joint motions reduce the number of solutions. If no solution is found, then the position is outside the manipulator reach.

The parameters k_1, k_2, and k_3 are used to find the inverse kinematic solution and should be specified by the user. However, in the case of a known set of joint angles, as in the case of the direct kinematics, these parameters can be computed, and consequently the arm configuration known, using simple equations derived in the next chapter.

2.3 Singular Points

As shown above, there are eight different solutions to the inverse kinematics depending on the arm configuration. Singular points (also called degenerate points or dead points) are the dividing boundaries

between the solution sets. They are also the points where the determinant of the Jacobian is zero and, as a result, there are several solutions for the end effector position.

There are two important properties of the singular points in connection with the control of the hand: (i) A loss in the number of degrees of freedom occurs. The kinematic equations become less accurate in the neighbourhood of a singular point and totally break down at the point. (ii) The singular points are the changing points of the kind of the configuration of the arm. Further discussion of singularities is given in ref. 5.

Featherstone (1) provides alternative equations that are accurate in a small neighbourhood around singular points. In this report, similar equations are derived to satisfy the PUMA 560 geometry and the results show good accuracy near the dead points.

CHAPTER 3. VELOCITY TRANSFORMATION

We present in this chapter algorithms for the transformation of velocities between the Cartesian space and the joint space for the PUMA 560 robot. The following algorithms assume that the intermediate and final results of the position transformation are available.

Note that the linear velocity $\underline{v}$ of the end effector with respect to the reference frame is equal to the time derivative of the position of the end effector $\underline{r}$.

$$\underline{v}(t) = \frac{d\,\underline{r}(t)}{dt}$$

The same is not true for the angular velocities $\underline{\omega}$. The instantaneous angular velocities of the end effector coordinate frame about the principal axes of the reference frame (6) are obtained as

$$\begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \begin{bmatrix} \cos\phi\cos\theta & -\sin\phi & 0 \\ \sin\phi\cos\theta & \cos\phi & 0 \\ -\sin\theta & 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{\psi} \\ \dot{\theta} \\ \dot{\phi} \end{bmatrix}$$

where ψ, θ, and ϕ are the roll (around x), pitch (around y), and yaw (around z) angles, respectively.

3.1 Cartesian-to-joint-space Velocity Tranformation

The problem is defined as follows: given the position/orientation of the end effector and its linear velocity $\underline{v}$ and angular velocity $\underline{\omega}$ with respect to the base, find $\dot{\underline{\theta}}$, the vector of joint rates.

The first step is to calculate the linear velocity of the wrist, from which the values of the first three joint rates can be calculated. To do so, one can follow Featherstone's steps exactly or alternatively use the method below. This method is more systematic than Featherstone's, especially when calculating the acceleration transformations.

Consider Fig. 4 where the first three joints and their associated coordinate systems are shown. The unit vectors $\underline{z}_1$, $\underline{z}_2$, and $\underline{z}_3$ are located along the links ℓ_1, ℓ_2, and ℓ_3, respectively. The coordinates of link 1 are rotated θ_1 around $\underline{z}_0$, the coordinates of link 2 are rotated θ_2 around x_1, and the coordinates of link 3 are rotated θ_3 around $\underline{x}_2$.

The position of the wrist with respect to the base $\underline{r}_w$ is defined by the following equation

$$[3.1] \qquad \underline{r}_w = \ell_1 \underline{z}_1 + d_1 (-\underline{x}_1) + \ell_2 \underline{z}_2 + d_2 \underline{x}_2 + \ell_3 \underline{z}_3$$

But since $\underline{x}_1 = \underline{x}_2$, and defining d as equal to the difference $(d_1 - d_2)$, we obtain

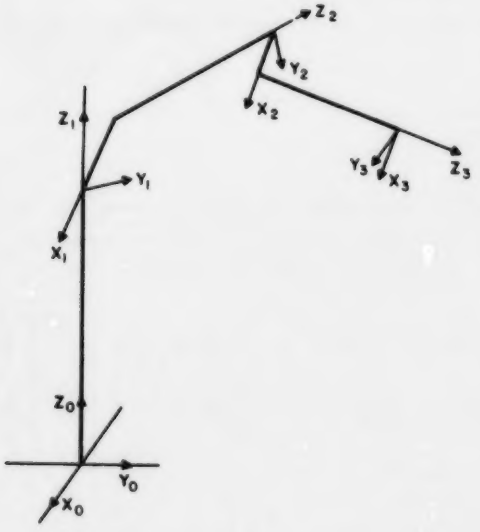


Figure 4. Coordinate systems for the first three joints of the PUMA 560 arm

[3.2] $$\underline{r}_w = \ell_1 \underline{z}_1 - d \underline{x}_1 + \ell_2 \underline{z}_2 + \ell_3 \underline{z}_3$$

To obtain the wrist's linear velocity, we differentiate equation [3.2] with respect to time.

[3.3] $$\underline{v}_w = \ell_1 \underline{\omega}_1 \times \underline{z}_1 - d \underline{\omega}_1 \times \underline{x}_1 + \ell_2 \underline{\omega}_2 \times \underline{z}_2 + \ell_3 \underline{\omega}_3 \times \underline{z}_3$$

where

$$\underline{\omega}_1 = \dot{\theta}_1 \underline{z}_1$$

$$\underline{\omega}_2 = \dot{\theta}_1 \underline{z}_1 - \dot{\theta}_2 \underline{x}_2$$

$$\underline{\omega}_3 = \dot{\theta}_1 \underline{z}_1 - (\dot{\theta}_2 + \dot{\theta}_3) \underline{x}_2$$

Substituting for ω_1, ω_2, and ω_3 in the terms of the right-hand side of equation [3.3], we obtain

$$\underline{\omega}_1 \times \underline{z}_1 = 0$$

$$\underline{\omega}_1 \times \underline{x}_1 = \dot{\theta}_1 (\underline{z}_1 \times \underline{x}_1) = \dot{\theta}_1 \underline{y}_1$$

$$\underline{\omega}_2 \times \underline{z}_2 = \dot{\theta}_1 (\underline{z}_1 \times \underline{z}_2) - \dot{\theta}_2 (\underline{x}_2 \times \underline{z}_2) = -\dot{\theta}_1 \sin \theta_2 \underline{x}_1 + \dot{\theta}_2 \underline{y}_2$$

$$\underline{\omega}_3 \times \underline{z}_3 = \dot{\theta}_1 (\underline{z}_1 \times \underline{z}_3) - (\dot{\theta}_2 + \dot{\theta}_3) (\underline{x}_2 \times \underline{z}_3)$$

$$= - \dot{\theta}_1 \sin (\theta_2 + \theta_3') \underline{x}_1 + (\dot{\theta}_2 + \dot{\theta}_3) \underline{y}_3$$

and equation [3.3] becomes

[3.4] $$\underline{v}_w = -d\dot{\theta}_1 \underline{y}_1 + \ell_2(-\dot{\theta}_1 \sin \theta_2 \underline{x}_1 + \dot{\theta}_2 \underline{y}_2) + \ell_3[-\dot{\theta}_1 \sin (\theta_2 + \theta_3') \underline{x}_1 + (\dot{\theta}_2 + \dot{\theta}_3) \underline{y}_3]$$

The linear velocity of the wrist in link 1 coordinates is obtained from equation [3.4] by replacing $\underline{y}_2$ and $\underline{y}_3$ with their equivalence in link 1 coordinates as follows

$$\underline{y}_2 = \cos \theta_2 \underline{y}_1 - \sin \theta_2 \underline{z}_1$$

[3.5] $$\underline{y}_3 = \cos (\theta_2 + \theta_3') \underline{y}_1 - \sin (\theta_2 + \theta_3') \underline{z}_1$$

$$^1\underline{v}_w = -\dot{\theta}_1 w_b \underline{x}_1 + [-d\dot{\theta}_1 + \dot{\theta}_2 w_a + \ell_3 \dot{\theta}_3 \cos (\theta_2 + \theta_3')] \underline{y}_1 - [\dot{\theta}_2 w_b + \ell_3 \dot{\theta}_3 \sin (\theta_2 + \theta_3')] \underline{z}_1$$

where

$$[3.6] \quad \begin{aligned} w_a &= \ell_2 \cos\theta_2 + \ell_3 \cos(\theta_2 + \theta_3') \\ w_b &= \ell_2 \sin\theta_2 + \ell_3 \sin(\theta_2 + \theta_3') \end{aligned}$$

But the wrist's linear velocity as a function of the desired end effector linear and angular velocities is

$$[3.7] \quad \underline{v}_w = \underline{v} + \underline{I}_4 \times \underline{\omega}$$

where $\underline{I}_4$ is the vector representing link ℓ_4 and its orientation.

$$[3.9] \quad \underline{I}_4 = -\ell_4 \sin r_\theta \sin r_\rho \, \underline{x}_0 + \ell_4 \sin r_\theta \cos r_\rho \, \underline{y}_0 + \ell_4 \cos r_\theta \, \underline{z}_0$$

Thus, the linear velocity of the wrist in the coordinates of link ℓ_1 is

$$[3.10] \quad {}^1\underline{v}_w = (\cos\theta_1 \, v_{wx} + \sin\theta_1 \, v_{wy}) \, \underline{x}_1 + (\cos\theta_1 \, v_{wy} - \sin\theta_1 \, v_{wx}) \, \underline{y}_1 + v_{wz} \, \underline{z}_1$$

The first three joint rates are obtained from equations [3.5] and [3.10] as

$$[3.11] \quad \dot{\theta}_1 = -{}^1v_{wx}/w_b$$

$$[3.12] \quad \dot{\theta}_2 = [(d\dot{\theta}_1 + {}^1v_{wy}) \sin(\theta_2 + \theta_3') + {}^1v_{wz} \cos(\theta_2 + \theta_3')]/\ell_2 \sin\theta_3'$$

$$[3.13] \quad \dot{\theta}_3 = -[(d\dot{\theta}_1 + {}^1v_{wy})w_b + w_a \, {}^1v_{wz}]/\ell_2\ell_3 \sin\theta_3'$$

The above equations deteriorate when $w_b \to 0$ and/or $\sin\theta_3' \to 0$ and totally break down when their value reaches zero. Thus when the robot is approaching a singularity, an alternative solution should take effect.

For the case where $w_b \to 0$ and the value of $\dot{\theta}_1$ becomes arbitrary. The sensible solution is to find $\dot{\theta}_1$ based on interpolation from neighbouring positions.

The second case is when link ℓ_2 and ℓ_3 are aligned, that is, $\sin\theta_3' = 0$, and equations [3.12] and [3.13] fail. Using equations [3.5] and [3.6] and substituting a zero value for θ_3', we obtain

$$\begin{aligned} {}^1v_{wy} &= -d\dot{\theta}_1 + (\ell_2 + \ell_3)\cos\theta_2 \, \dot{\theta}_2 + \ell_3 \cos\theta_2 \, \dot{\theta}_3 \\ {}^1v_{wz} &= -(\ell_2 + \ell_3)\sin\theta_2 \, \dot{\theta}_2 - \ell_3 \sin\theta_2 \, \dot{\theta}_3 \end{aligned}$$

from which the following relation between $\dot{\theta}_2$ and $\dot{\theta}_3$ is obtained.

$$[3.14] \quad (\ell_2 + \ell_3)\dot{\theta}_2 + \ell_3\dot{\theta}_3 = \cos\theta_2\,({}^1v_{wy} + d_1\dot{\theta}_1) - \sin\theta_2\,{}^1v_{wz}$$

Equation [3.14] is approximate as θ_3' approaches zero and exact for $\theta_3' = 0$. To find the value of both joints' rates, $\dot{\theta}_2$ and $\dot{\theta}_3$, the value of either of them should be found by interpolation and then equation [3.14] used to find the other.

The last three joint rates are calculated using the equations developed in ref. 1 and referred to here by their corresponding equation number and letter F beside it. Let ${}^1\omega_{ew}$ be the angular velocity of the end effector with respect to the wrist (i.e., the desired angular velocity $\underline{\omega}$ minus the angular velocity due to the first three joints) expressed in the coordinate of link ℓ_1; then

$$[29F] \quad {}^1\omega_{ew} = (\cos\theta_1\,\omega_x + \sin\theta_1\,\omega_y + \dot{\theta}_2 + \dot{\theta}_3)\,\underline{x}_1 + (\cos\theta_1\,\omega_y - \sin\theta_1\,\omega_x)\,\underline{y}_1 + (\omega_z - \dot{\theta}_1)\,\underline{z}_1$$

Now expressing ω_{ew} in link 3 coordinates

$$[30F] \quad {}^3\omega_{ew} = {}^1\omega_{ewx}\,\underline{x}_3 + [\cos(\theta_2 + \theta_3)\,{}^1\omega_{ewy} - \sin(\theta_2 + \theta_3)\,{}^1\omega_{ewz}]\,\underline{y}_3 + [\cos(\theta_2 + \theta_3)\,{}^1\omega_{ewz} + \sin(\theta_2 + \theta_3)\,{}^1\omega_{ewy}]\,\underline{z}_3$$

But ${}^3\omega_{ew}$ can be expressed as a function of the joints' rates as follows

$$[31F] \quad {}^3\omega_{ew} = \dot{\theta}_4\,\underline{z}_3 - \dot{\theta}_5\,(\cos\theta_4\,\underline{x}_3 + \sin\theta_4\,\underline{y}_3) + \dot{\theta}_6\,(-\sin\theta_4\sin\theta_5\,\underline{x}_3 + \cos\theta_4\sin\theta_5\,\underline{y}_3 + \cos\theta_5\,\underline{z}_3)$$

That is,

$$[32F] \quad \begin{bmatrix} 0 & -\cos\theta_4 & -\sin\theta_4\sin\theta_5 \\ 0 & -\sin\theta_4 & \cos\theta_4\sin\theta_5 \\ 1 & 0 & \cos\theta_5 \end{bmatrix} \begin{bmatrix} \dot{\theta}_4 \\ \dot{\theta}_5 \\ \dot{\theta}_6 \end{bmatrix} = \begin{bmatrix} {}^3\omega_{ewx} \\ {}^3\omega_{ewy} \\ {}^3\omega_{ewz} \end{bmatrix}$$

Solving [32F] gives

$$[33F] \quad \dot{\theta}_5 = -\cos\theta_4\,{}^3\omega_{ewx} - \sin\theta_4\,{}^3\omega_{ewy}$$

[34F] $$\dot{\theta}_6 = (\cos\theta_4\ {}^3\omega_{ewy} - \sin\theta_4\ {}^3\omega_{ewx})/\sin\theta_5$$

[35F] $$\dot{\theta}_4 = {}^3\omega_{ewz} - \cos\theta_5\ \dot{\theta}_6$$

Note that the accuracy of these equations deteriorates as $\sin\theta_5 \to 0$ and breaks down completely when $\sin\theta_5 = 0$.

For the joint rates $\dot{\theta}_6$ and $\dot{\theta}_4$ to be achievable, the numerator of equation [34F] should equal to zero. If this is the case, one of the joint rates should be chosen and the other computed using equation [35F].

This transformation method is considerably faster than any method based on the Jacobian matrices.

3.2 Joint-space-to-Cartesian Velocity Transformation

This is the reverse problem of the previous section. The problem is defined as follows: given the joint angles $\underline{\theta}$ and the joint rates $\dot{\underline{\theta}}$ find the linear velocity $\underline{v}$ and the angular velocity $\underline{\omega}$ of the end effector.

To solve this problem, the procedure outlined in the previous section to find the joint rates may be easily reversed.

First, equation [32F] is used to find the values of ${}^3\underline{\omega}_{ew}$, the angular velocity of the end effector with respect to the wrist expressed in link ℓ_3 coordinates. Then, inverting equations [30F] and [29F], the angular velocity $\underline{\omega}$ is obtained as follows:

[3.15] $$\begin{aligned} {}^1\omega_{ew} = {}^3\omega_{ewx}\,\underline{x}_1 &+ [\cos(\theta_2+\theta_3)\ {}^3\omega_{ewy} + \sin(\theta_2+\theta_3)\ {}^3\omega_{ewz}]\,\underline{y}_1 \\ &+ [\cos(\theta_2+\theta_3)\ {}^3\omega_{ewz} - \sin(\theta_2+\theta_3)\ {}^3\omega_{ewy}]\,\underline{z}_1 \end{aligned}$$

and

[3.16] $$\begin{aligned} \underline{\omega} = &[\cos\theta_1\ ({}^1\omega_{ewx} - \dot{\theta}_2 - \dot{\theta}_3) - \sin\theta_1\ {}^1\omega_{ewy}]\,\underline{x}_0 \\ &+ [\cos\theta_1\ {}^1\omega_{ewy} + \sin\theta_1\ ({}^1\omega_{ewx} - \dot{\theta}_2 - \dot{\theta}_3)]\,\underline{y}_0 \\ &+ ({}^1\omega_{ewz} + \dot{\theta}_1)\,\underline{z}_0 \end{aligned}$$

The linear velocity $\underline{v}$ can be calculated from equation [3.7] as

[3.17] $$\underline{v} = \underline{v}_w + \underline{\omega} \times \underline{I}_4$$

The velocity $\underline{v}_w$ of the wrist with respect to the base can be calculated by inverting equation [3.10]. The value of ${}^1\underline{v}_w$, the velocity of the wrist expressed in link ℓ_1 coordinates, needed in equation [3.10], is obtained from equation [3.11].

$$[3.18] \quad \underline{v} = (\cos\theta_1\ {}^1v_{wx} - \sin\theta_1\ {}^1v_{wy})\ \underline{x}_0 + (\cos\theta_1\ {}^1v_{wy} + \sin\theta_1\ {}^1v_{wx})\ \underline{y}_0 + {}^1v_{wz}\ \underline{z}_0 + \underline{\omega} \times \underline{I}_4$$

This procedure always works even at the singularities and the accuracy is good everywhere. Comparing this procedure to the general method (the two are quite similar), it is noticed that the increase in speed is not as great as that realized in the previous section.

CHAPTER 4. ACCELERATION TRANSFORMATION

In this chapter, we present the algorithms for the transformation of accelerations between the Cartesian space and the joint space for the PUMA 560 robot. The following algorithms assume that the joint angles and rates and the Cartesian position and velocities are known and that any needed intermediate results are available.

4.1 Cartesian-to-joint-space Acceleration Transformation

The problem is defined as follows: given the position/orientation of the end effector, its linear and angular velocities $\underline{v}$ and $\underline{\omega}$, and its linear and angular accelerations $\underline{a}$ and $\underline{\alpha}$, find $\ddot{\underline{\theta}}$, the vector of joint acceleration

The first step is to calculate the linear acceleration of the wrist $\underline{a}_w$ from which the values of the first three joints' accelerations can be calculated. The linear acceleration of the end effector $\underline{a}$ can be expressed as:

$$[4.1] \quad \underline{a} = \underline{a}_w + \underline{\omega} \times (\underline{\omega} \times \underline{I}_4) + \underline{\alpha} \times \underline{I}_4$$

from which $\underline{a}_w$ can be found since all other terms are either given or calculated previously. Then the linear acceleration of the wrist ${}^1\underline{a}_w$, expressed in link ℓ_1 coordinates is

$$[4.2] \quad {}^1\underline{a}_w = (\cos\theta_1\ a_{wx} + \sin\theta_1\ a_{wy})\ \underline{x}_1 + (\cos\theta_1\ a_{wy} - \sin\theta_1\ a_{wx})\ \underline{y}_1 + a_{wz}\ \underline{z}_1$$

But the vector ${}^1\underline{a}_w$ can be calculated using the method outlined in the previous section: the linear velocity expressed in the coordinates of link

ℓ_1 equation [3.5] is differentiated to obtain the linear acceleration.

$$[4.3]\quad {}^1\underline{a}_w = -\ddot{\theta}_1 w_b\,\underline{x}_1 - \dot{\theta}_1\dot{w}_b\,\underline{x}_1 - \dot{\theta}_1 w_b\,\underline{\omega}_1 \times \underline{x}_1$$
$$+ [-d\ddot{\theta}_1 + \ddot{\theta}_2 w_a + \dot{\theta}_2\dot{w}_a + \ell_3\ddot{\theta}_3 \cos(\theta_2 + \theta_3') - \ell_3\dot{\theta}_3(\dot{\theta}_2 + \dot{\theta}_3)\sin(\theta_2 + \theta_3')]\,\underline{y}_1$$
$$+ [-d\dot{\theta}_1 + w_a\dot{\theta}_2 + \ell_3\dot{\theta}_3 \cos(\theta_2 + \theta_3')]\,\underline{\omega}_1 \times \underline{y}_1$$
$$+ [\ddot{\theta}_2 w_b + \dot{\theta}_2\dot{w}_b + \ell_3\ddot{\theta}_3 \sin(\theta_2 + \theta_3') + \ell_3\dot{\theta}_3(\dot{\theta}_2 + \dot{\theta}_3)\cos(\theta_2 + \theta_3')]\,\underline{z}_1$$
$$- [\dot{\theta}_2 w_b + \ell_3\dot{\theta}_3 \sin(\theta_2 + \theta_3')]\,\underline{\omega}_1 \times \underline{z}_1$$

But

$$\dot{w}_a = -\dot{\theta}_2 w_b - \ell_3\dot{\theta}_3 \sin(\theta_2 + \theta_3')$$

$$\dot{w}_b = \dot{\theta}_2 w_a + \ell_3\dot{\theta}_3 \cos(\theta_2 + \theta_3')$$

$$\underline{\omega}_1 \times \underline{x}_1 = \dot{\theta}_1\,\underline{y}_1$$

$$\underline{\omega}_1 \times \underline{y}_1 = -\dot{\theta}_1\,\underline{x}_1$$

$$\underline{\omega}_1 \times \underline{z}_1 = 0$$

Using these relationships in equation [4.3], the following components of ${}^1a_w$ are obtained

$$[4.4]\quad {}^1a_{wx} = -\ddot{\theta}_1 w_b + f_{x1}$$

$$[4.5]\quad {}^1a_{wy} = -d\ddot{\theta}_1 + w_a\ddot{\theta}_2 + \ell_3\ddot{\theta}_3 \cos(\theta_2 + \theta_3') + f_{x2}$$

$$[4.6]\quad {}^1a_{wz} = -\ddot{\theta}_2 w_b - \ell_3\ddot{\theta}_3 \sin(\theta_2 + \theta_3') + f_{x3}$$

or in a matrix form

$$[4.7]\quad {}^1a_w = \begin{bmatrix} -w_b & 0 & 0 \\ -d & w_a & \ell_3\cos(\theta_2 + \theta_3') \\ 0 & -w_b & -\ell_3\sin(\theta_2 + \theta_3') \end{bmatrix} \begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \\ \ddot{\theta}_3 \end{bmatrix} + \begin{bmatrix} f_{x1} \\ f_{y1} \\ f_{z1} \end{bmatrix}$$

where

$$f_{x1} = d\dot{\theta}_1^2 - 2\dot{\theta}_1\dot{\theta}_2 w_a - 2\ell_3\dot{\theta}_1\dot{\theta}_3 \cos(\theta_2 + \theta_3')$$

$$f_{x2} = -(\dot{\theta}_1^2 + \dot{\theta}_2^2) w_b - \ell_3\dot{\theta}_3(2\dot{\theta}_2 + \dot{\theta}_3) \sin(\theta_2 + \theta_3')$$

$$f_{x3} = -\dot{\theta}_2^2 w_a - \ell_3\dot{\theta}_3(2\dot{\theta}_2 + \dot{\theta}_3) \cos(\theta_2 + \theta_3')$$

The first three joints' accelerations are found by inverting equation [4.7].

[4.8] $$\ddot{\theta}_1 = -({}^1a_{wx} - f_{x1})/w_b$$

[4.9] $$\ddot{\theta}_2 = \{\sin(\theta_2 + \theta_3')[+ d\ddot{\theta}_1 + ({}^1a_{wy} - f_{x2})] + \cos(\theta_2 + \theta_3')({}^1a_{wz} - f_{x3})\}/\ell_2 \sin\theta_3'$$

[4.10] $$\ddot{\theta}_3 = [d({}^1a_{wx} - f_{x1}) - w_b({}^1a_{wy} - f_{x2}) - w_a({}^1a_{wz} - f_{x3})]/\ell_2\ell_3 \sin\theta_3'$$

Examining equations [4.8] to [4.10], it is noticed that their solution deteriorates as w_b and/or θ_3' approaches zero and totally breaks down when their value reaches zero.

If $w_b = 0$, $\ddot{\theta}_1$ and $\ddot{\theta}_2$ are arbitrary and a choice of the value of either should be based on interpolation from neighbouring positions. The value of the other joint acceleration should be calculated using equation [4.9].

If $\theta_3' = 0$, equations [4.9] and [4.10] fail. Substituting for w_a and w_b in equations [4.5] and [4.6] and setting $\theta_3' = 0$, we obtain

[4.11] $${}^1a_{wy} + d\ddot{\theta}_1 + \dot{\theta}_1^2(\ell_2 + \ell_3)\sin\theta_2 = \cos\theta_2[(\ell_2 + \ell_3)\ddot{\theta}_2 + \ell_3\ddot{\theta}_3] - \sin\theta_2[(\ell_2 + \ell_3)\dot{\theta}_2^2 + \ell_3\dot{\theta}_3(2\dot{\theta}_2 + \dot{\theta}_3)$$

[4.12] $${}^1a_{wz} = -\sin\theta_2[(\ell_2 + \ell_3)\ddot{\theta}_2 + \ell_3\ddot{\theta}_3] - \cos\theta_2[(\ell_2 + \ell_3)\dot{\theta}_2^2 + \ell_3\dot{\theta}_3(2\dot{\theta}_2 + \dot{\theta}_3)]$$

Adding $\cos\theta_2$ times equation [4.11] to $-\sin\theta_2$ times equation [4.12] we obtain the equation relating $\ddot{\theta}_2$ and $\ddot{\theta}_3$ as

[4.13] $$(\ell_2 + \ell_3)\ddot{\theta}_2 + \ell_3\ddot{\theta}_3 = \cos\theta_2[{}^1a_{wy} + d\ddot{\theta}_1 + \dot{\theta}_1^2(\ell_2 + \ell_3)\sin\theta_2] - \sin\theta_2\,{}^1a_{wz}$$

Either $\ddot{\theta}_2$ or $\ddot{\theta}_3$ should be chosen by interpolation and the other evaluated using equation [4.13].

The first three joints' accelerations have determined the linear acceleration of the wrist. The remaining accelerations ($\ddot{\theta}_4$, $\ddot{\theta}_5$, and $\ddot{\theta}_6$) have to determine the angular acceleration $\underline{\alpha}$ of the end effector with respect to the wrist. Let $\underline{\omega}_{ew}$ be the angular velocity of the end effector with respect to the wrist, then

$$[4.14] \quad \underline{\omega}_{ew} = \dot{\theta}_4 \underline{z}_3 - \dot{\theta}_5 \underline{x}_4 + \dot{\theta}_6 \underline{z}_5$$

Examining equation [4.14], it is logical to solve in terms of link ℓ_4 coordinates where

$$\underline{z}_3 = \underline{z}_4$$

$$\underline{z}_5 = \cos\theta_5 \underline{z}_4 + \sin\theta_5 \underline{y}_4$$

Thus the angular acceleration of the end effector with respect to the wrist, $\underline{\alpha}_{ew}$, expressed in link ℓ_4 coordinates, is

$$[4.15] \quad {}^4\underline{\alpha}_{ew} = (-\ddot{\theta}_5 - \dot{\theta}_4\dot{\theta}_6 \sin\theta_5)\, \underline{x}_4 + (\sin\theta_5 \ddot{\theta}_6 + \cos\theta_5 \dot{\theta}_5\dot{\theta}_6 - \dot{\theta}_4\dot{\theta}_5)\, \underline{y}_4 + (\ddot{\theta}_4 + \cos\theta_5 \ddot{\theta}_6 - \sin\theta_5 \dot{\theta}_5\dot{\theta}_6)\, \underline{z}_4$$

The value of ${}^4\underline{\alpha}_{ew}$ as a function of the desired end effector acceleration is found by first expressing $\underline{\alpha}_{ew}$ in link ℓ_1 coordinates. Since

$${}^1\underline{\omega} = {}^1\underline{\omega}_w + {}^1\underline{\omega}_{ew}$$

then

$${}^1\underline{\alpha} = {}^1\dot{\underline{\omega}}_w + {}^1\dot{\underline{\omega}}_{ew} + {}^1\underline{\omega}_w \times {}^1\underline{\omega}_{ew}$$

and it follows that

$$[4.16] \quad {}^1\underline{\alpha}_{ew} = {}^1\dot{\underline{\omega}}_{ew} = {}^1\underline{\alpha} - {}^1\dot{\underline{\omega}}_w - {}^1\underline{\omega}_w \times {}^1\underline{\omega}_{ew}$$

But

$${}^1\underline{\omega}_w = -(\dot{\theta}_2 + \dot{\theta}_3)\, \underline{i} + \dot{\theta}_1 \underline{k}$$

[4.17] $$^1\dot{\omega}_w = (\ddot{\theta}_2 + \ddot{\theta}_3)\,\underline{i} + \dot{\theta}_1(\dot{\theta}_2 + \dot{\theta}_3)\,\underline{j} - \ddot{\theta}_1\,\underline{k}$$

$$^1\underline{\omega}_{ew} = {}^1\underline{\omega} - {}^1\underline{\omega}_w$$

$$^1\underline{\omega} = (\cos\theta_1\,\omega_x + \sin\theta_1\,\omega_y)\,\underline{i} + (\cos\theta_1\,\omega_y - \sin\theta_1\,\omega_x)\,\underline{j} + \omega_z\,\underline{k}$$

and

[4.18] $$^1\underline{\omega}_w \times {}^1\underline{\omega}_{ew} = -\dot{\theta}_1\,(\cos\theta_1\,\omega_y - \sin\theta_1\,\omega_x)\,\underline{i}$$
$$+ [\dot{\theta}_1\,(\cos\theta_1\,\omega_x + \sin\theta_1\,\omega_y + (\dot{\theta}_2 + \dot{\theta}_3)\,\omega_z]\,\underline{j}$$
$$-(\dot{\theta}_2 + \dot{\theta}_3)\,(\cos\theta_1\,\omega_y - \sin\theta_1\,\omega_x)\,\underline{k}$$

Substituting the given value of $\underline{\alpha}$ and equations [4.17] and [4.18] into [4.16], $^1\underline{\alpha}_{ew}$ is obtained.

$$^1\alpha_{ewx} = \cos\theta_1\,\alpha_x + \sin\theta_1\,\alpha_y + \ddot{\theta}_2 + \ddot{\theta}_3 + \dot{\theta}_1(\cos\theta_1\,\omega_y - \sin\theta_1\,\omega_x)$$

[4.19] $$^1\alpha_{ewy} = \cos\theta_1\,\alpha_y - \sin\theta_1\,\alpha_x - \dot{\theta}_1(\dot{\theta}_2 + \dot{\theta}_3)$$
$$-\dot{\theta}_1\,(\cos\theta_1\,\omega_x + \sin\theta_1\,\omega_y) - (\dot{\theta}_2 + \dot{\theta}_3)\,\omega_z$$

$$^1\alpha_{ewz} = \alpha_z - \ddot{\theta}_1 + (\dot{\theta}_2 + \dot{\theta}_3)\,(\cos\theta_1\,\omega_y - \sin\theta_1\,\omega_x)$$

And consequently

$$^4\alpha_{ewx} = {}^1\alpha_{ewx}\cos\theta_4 + [\cos(\theta_2 + \theta_3)\,{}^1\alpha_{ewy}$$
$$-\sin(\theta_2 + \theta_3)\,{}^1\alpha_{ewz}]\sin\theta_4$$

[4.20] $$^4\alpha_{ewy} = -{}^1\alpha_{ewx}\sin\theta_4 + [\cos(\theta_2 + \theta_3)\,{}^1\alpha_{ewy}$$
$$-\sin(\theta_2 + \theta_3)\,{}^1\alpha_{ewz}]\cos\theta_4$$

$$^4\alpha_{ewz} = \cos(\theta_2 + \theta_3)\,{}^1\alpha_{ewz} + \sin(\theta_2 + \theta_3)\,{}^1\alpha_{ewy}$$

From equations [4.15] and [4.20], the values of the last three joints' accelerations are

[4.21] $$\ddot{\theta}_5 = -{}^4\alpha_{ewx} - \dot{\theta}_4\dot{\theta}_6\sin\theta_5$$

[4.22] $$\ddot{\theta}_6 = ({}^4\alpha_{ewy} + \dot{\theta}_4\dot{\theta}_5 - \dot{\theta}_5\dot{\theta}_6\cos\theta_5)/\sin\theta_5$$

[4.23] $$\ddot{\theta}_4 = {}^4\alpha_{ewz} + \dot{\theta}_5\dot{\theta}_6\sin\theta_5 - \theta_6\cos\theta_5$$

The accuracy of equation [4.22] deteriorates as $\sin\theta_5 \to 0$ and these equations cannot be solved at $\sin\theta_5 = 0$. For the joints' accelerations $\ddot{\theta}_4$ and $\ddot{\theta}_5$ to be achievable, the numerator of $\ddot{\theta}_6$ should also equal to zero. If this is the case one of the joint accelerations ($\ddot{\theta}_4$ or $\ddot{\theta}_6$) should be chosen arbitrarily and the other computed using equation [4.23].

4.2 Joint-space-to-Cartesian Transformation

This is the reverse problem of the previous section. The problem is defined as follows: given the joint angles $\underline{\theta}$, joint rates $\underline{\dot{\theta}}$, and joint accelerations $\underline{\ddot{\theta}}$, find the linear acceleration $\underline{a}$ and the angular acceleration $\underline{\alpha}$ of the end effector.

To solve this problem, the procedure outlined in the previous section may easily be reversed. First, equation [4.15] is used to find ${}^4\underline{\alpha}_{ew}$, then equations [4.20] and [4.19] are used to find the end effector angular acceleration $\underline{\alpha}$ as follows:

$$
\begin{aligned}
{}^1\alpha_{ewx} &= \cos\theta_4\ {}^4\alpha_{ewx} - \sin\theta_4\ {}^4\alpha_{ewy} \\
{}^1\alpha_{ewy} &= \cos(\theta_2+\theta_3)\,(\sin\theta_4\ {}^4\alpha_{ewx} + \cos\theta_4\ {}^4\alpha_{ewy}) \\
&\qquad + \sin(\theta_2+\theta_3)\ {}^4\alpha_{ewz} \\
{}^1\alpha_{ewz} &= -\sin(\theta_2+\theta_3)\,(\sin\theta_4\ {}^4\alpha_{ewx} + \cos\theta_4\ {}^4\alpha_{ewy}) \\
&\qquad + \cos(\theta_2+\theta_3)\ {}^4\alpha_{ewz}
\end{aligned}
\qquad [4.24]
$$

and

$$
\begin{aligned}
\underline{\alpha} = {} & (\cos\theta_1 f_x - \sin\theta_1 f_y)\,\underline{x}_0 + (\sin\theta_1 f_x + \cos\theta_1 f_y)\,\underline{y}_0 \\
& + [{}^1\alpha_{ewz} + \ddot{\theta}_1 + (\dot{\theta}_2+\dot{\theta}_3)(\cos\theta_1\omega_y - \sin\theta_1\omega_x)]\underline{z}_0
\end{aligned}
\qquad [4.25]
$$

where

$$
\begin{aligned}
f_x &= {}^1\alpha_{ewx} - \ddot{\theta}_2 - \ddot{\theta}_3 - \dot{\theta}_1(\cos\theta_1\omega_y - \sin\theta_1\omega_x) \\
f_y &= {}^1\alpha_{ewy} + \dot{\theta}_1(\dot{\theta}_2+\dot{\theta}_3) + \dot{\theta}_1(\cos\theta_1\omega_x + \sin\theta_1\omega_y) \\
&\qquad + (\dot{\theta}_2+\dot{\theta}_3)\,\omega_z
\end{aligned}
$$

Once $\underline{\alpha}$ is found, equation [4.7] is used to find ${}^1\underline{a}_w$ followed by equation [4.1] to find $\underline{a}$ as follows

$$\underline{a} = \underline{a}_w + \underline{\omega} \times (\underline{\omega} \times \underline{I}_4) + \underline{\alpha} \times \underline{I}_4$$

where

$$\underline{a}_w = (\cos\theta_1\ {}^1a_{wx} - \sin\theta_1\ {}^1a_{wy})\ \underline{i} + (\sin\theta_1\ {}^1a_{wx} + \cos\theta_1\ {}^1a_{wy})\ \underline{j} + {}^1a_{wz}\ \underline{k}$$

This procedure always works even at singularities.

CHAPTER 5. SUMMARY AND DISCUSSION

Efficient algorithms for the calculation of the direct and inverse kinematic velocities and accelerations for the PUMA 560 robot have been presented. These algorithms are based on a method developed by Featherstone (1) for the calculation of kinematic positions and velocities for a coplanar arm with a spherical wrist. The algorithms exploit the simple geometry of the arm. The structure of the spherical wrist allows a decomposition the six-degree-of-freedom problem into two three-degree-of-freedom problems representing position and orientation.

This solution approach develops a good physical understanding for the manipulator and provides geometric insight into the transformation problem, but is dependent upon the particular manipulator for which the solution is desired. For any six-degree-of-freedom robot with a spherical wrist, the solution of the first three joints depends on the given robot, whereas the solution of the last three joints (forming the spherical wrist) is independent of the robot.

The velocity transformation cannot be directly compared to the method of Paul et al. (7) because the Cartesian velocity specifications are different. Hollerback and Sahar (3) compared both methods for the Stanford arm using similar specifications and concluded that Featherstone's method is almost three times more efficient than Paul's method. Referring to section 6 of Featherstone (2), the inverse velocity transformation for the coplanar arm is considerably faster (26a + 48m + 10f) than any method based on the Jacobian matrices.

For the algorithms described in this paper, the total number of operations required is: (i) 21a + 30m + 0f and 23a + 35m + 6f for the direct and inverse velocity transformations, assuming that position transformations have already been performed, and (ii) 44a + 52m + 0f and 45a + 59m + 0f for direct and inverse acceleration transformations, assuming that the position and velocity transformations have already been performed.

The advantage of the present method is in the efficiency of calculation of the velocity and acceleration transformations. In order to find the kinematic velocities/accelerations, one uses intermediate and final results of the kinematic positions/velocities. For the accurate control of the arm motion all these solutions are needed, resulting in a substantial saving in computation time, a critical consideration for real time control.

REFERENCES

1. Featherstone, R. Position and velocity transformations between robot end-effector coordinates and joint angles. Int. J. Robot. Res. 2(2): 35-45; 1983

2. Featherstone, R. Calculation of robot joint rates and actuator torques from end effector velocities and applied forces. Mech. Mach. Theory, 18(3): 193-198; 1983.

3. Hollerback, J.M., and Sahar, G. Wrist-partitioned, inverse kinematic accelerations and manipulator dynamics. Int. J. Robot. Res. 2(4): 61-76; 1983.

4. Elgazzar, S. Efficient solution for the kinematic positions for the PUMA 560 robot. NRC/ERB-973, December 1984. 36 pages.

5. Uchiyama, M. A study of computer control of motion of a mechanical arm. Bull. Jpn. Soc. Mech. Eng. 22(173): 1640-1647. 1979

6. Frazer, R.A., Duncan, W.J., and Colla, A.R. Elementary matrices and some applications to dynamics and differential equations. Cambridge University Press. 1957. pp. 246-256.

7. Paul, R.P., Shimano, B., and Mayer, G.E. Differential kinematic control equations for simple manipulators. IEEE Trans. Syst., Man Cyber. SMC-11(6): 456-460; 1981.

APPENDIX A. SOLUTION FOR KINEMATIC POSITIONS

The derivation of the equations and the solution for the kinematic positions can be found in Elgazzar (4). Only the final results are listed here.

A.1 Direct Kinematic Solution

The problem is stated as follows: given the vector of joint angles $\underline{\theta} = (\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6)$, find the vector of position/orient $\underline{R} = (r_x, r_y, r_z, r_\rho, r_\theta, r_\psi)$. The solution vector $\underline{R}$ is

$$\cos r_\theta = \cos(\theta_2 + \theta_3)\cos\theta_5 - \sin(\theta_2 + \theta_3)\sin\theta_5\cos\theta_4$$

$$r_\psi = \theta_6 + \text{atan 2}\,[\sin(\theta_2 + \theta_3)\sin\theta_4,\ \sin\theta_5\cos(\theta_2 + \theta_3) + \cos\theta_5\sin(\theta_2 + \theta_3)\cos\theta_4]$$

$$r_\rho = \theta_1 + \text{atan 2}\,[\sin\theta_5\sin\theta_4,\ \sin(\theta_2 + \theta_3)\cos\theta_5 + \cos(\theta_2 + \theta_3)\sin\theta_5\cos\theta_4]$$

$$r_x = -w_b\sin\theta_1 - d\cos\theta_1 - \ell_4\sin r_\theta\sin r_\rho$$

$$r_y = w_b\cos\theta_1 - d\sin\theta_1 + \ell_4\sin r_\theta\cos r_\rho$$

$$r_z = w_a + \ell_1 + \ell_4\cos r_\theta$$

where

$$w_a = \ell_2\cos\theta_2 + \ell_3\cos(\theta_2 + \theta_3')$$

$$w_b = \ell_2\sin\theta_2 + \ell_3\sin(\theta_2 + \theta_3')$$

ℓ_1, ℓ_2, ℓ_3 and ℓ_4 length of links 1, 2, 3, and 4, respectively

d net offset in the arm due to shoulder offset d_1 and elbow offset d_2

θ_3' as defined in chapter 2

A.2 Inverse Kinematic Solution

The inverse kinematic problem is the following:

given the Cartesian position/orientation vector $\underline{R} = (r_x, r_y, r_z, r_\rho, r_\theta, r_\psi)$ and the arm configuration parameters k_1, k_2, and k_3, find the joint angles vector $\underline{\theta} = (\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6)$

The solution vector $\underline{\theta}$ is as follows:

$$\theta_1 = \text{atan2}\ (-k_1 w_{1x},\ k_1 w_{1y}) - k_1\ \text{atan2}\ (d,\ \ell)$$

where

$$\ell = (w_{1x}^2 + w_{1y}^2 - d^2)^{1/2}$$

and $\underline{w}_1 = (w_{1x},\ w_{1y},\ w_{1z})$ is the position of the wrist with respect to the base coordinate.

$$\theta_3 = k_1\ k_2\ \theta_3' + \delta$$

where

$$\cos\theta_3' = \frac{n^2 - \ell_2^2 - \ell_3^2}{2\ell_2\ell_3}$$

$$n = (-w_b \sin\theta_1,\ w_b \cos\theta_1,\ w_a)$$

and ρ as defined in chapter 2.

$$\theta_2 = k_1[\text{atan2}\ (\ell,\ w_{1z} - \ell_1) - k_2\ \text{atan2}\ (\ell_3 \sin\theta_3',\ \ell_2 + \ell_3 \cos\theta_3')]$$

$$\cos\theta_5 = \cos(\theta_2 + \theta_3)\cos r_\theta + \sin(\theta_2 + \theta_3)\sin r_\theta \cos(r_\rho - \theta_1)$$

$$\theta_4 = \text{atan2}\ [\sin r_\theta \sin(r_\rho - \theta_1),\ \cos(\theta_2 + \theta_3)\sin r_\theta \cos(r_\rho - \theta_1) - \sin(\theta_2 + \theta_3)\cos r_\theta]$$

$$\theta_6 = r_\psi - \text{atan2}\ [\sin(\theta_2 + \theta_3)\sin(r_\rho - \theta_1),\ \sin r_\theta \cos(\theta_2 + \theta_3) - \cos r_\theta \sin(\theta_2 + \theta_3)\cos(r_\rho - \theta_1)]$$

APPENDIX B. PROGRAM LISTINGS

The programs listed in this appendix 'kinempva.e' represents the algorithms for the combined position, velocity, and acceleration transformation. Also listed are the arctan function in 'atan2.c' and the constants in 'defin.h'.

The C-function 'jt-cart' solves for the direct kinematics while the function 'cart-jt' solves for the inverse kinematics. The main program 'mainkpva.c' reads the type of transformation to be performed. In the case of a direct transformation main accepts the joint angles either as referenced by Unimation and transform them to the format defined in this report or accepts them directly as defined here.

In the case of the inverse kinematics, main accepts the Cartesian coordinates either in the PUMA format (x_p, y_p, z_p, O, A, T) or using for the orientation of the arm, either the Euler angles (ϕ_e, θ_e, ψ_e) or the rotation angles (r_ρ, r_θ, r_ψ). The main program transforms the orientation to its equivalent (r_ρ, r_θ, r_ψ) if different from it. The solution is then outputted in the same input format.

The main program accepts the angles, the angular velocities and the angular accelerations in degrees, degrees/s and degrees/s^2, respectively. Also the positions, the linear velocities, and the linear accelerations are accepted as mm, mm/s, and mm/s^2.

```
                    /*   mainkpva.c   */


      #include <std.h>
      #include "definva.h"
      #include "atan.c"
      #include "kinempva.c"

      extern float rx=0,ry=0,rz=0,rho=0,rtheta=0,reps=0;
      extern float th1=0,th2=0,th3=0,th4=0,th5=0,th6=0;
      extern float th1d=0,th2d=0,th3d=0,th4d=0,th5d=0,th6d=0;
      extern float vx=0,vy=0,vz=0,wx=0,wy=0,wz=0;
      extern float th1dd=0,th2dd=0,th3dd=0,th4dd=0,th5dd=0,th6dd=0;
      extern float ax=0,ay=0,az=0,wdx=0,wdy=0,wdz=0;
      extern int status=0,k1=1,k2=1,k3=1;


main()

      /*  Reads either Cartesian coord. of a point (mm , mm/s, mm/s^2,
      degrees, degrees/s and degrees/s^2 ) or joint coord. (degrees,
      degrees/s, degrees/s^2 ), and does the necessary transformations.

           The input variables are:
             euler = 1 ---> orientation expressed in euler angles and
                            joint angles referenced as in report
                   = 0      otherwise
              oat  = 1 ---> orientation expressed in O, A, T angles and
                            joint angles referenced to the PUMA
                   = 0      otherwise
          jt_to_cart = 1 ---> transform is from jt coord. to cart. coord.
                     = 0    transform is from cart. coord. to jt coord. */


{
      extern float th1,th2,th3,th4,th5,th6;
      extern float th1d,th2d,th3d,th4d,th5d,th6d;
      extern float th1dd,th2dd,th3dd,th4dd,th5dd,th6dd;

      extern float rx,ry,rz,rho,rtheta,reps;
      extern float vx,vy,vz,wx,wy,wz;
      extern float ax,ay,az,wdx,wdy,wdz;
      extern int status,k1,k2,k3;
      int oat,euler,jt_to_cart;
```

```
float o,a,t,ephi,etheta,eeps;

putfmt (" euler =  n");
getfmt ("%i",&euler);
putfmt (" oat =  n");
getfmt ("%i",&oat);
putfmt (" jt_to_cart =  n");
getfmt ("%i",&jt_to_cart);

if ( jt_to_cart == 1 )
{
     putfmt (" th1 to th6 ; one per line (8.3) n");
     getfmt ("%8.5f",&th1);
     getfmt ("%8.5f",&th2);
     getfmt ("%8.5f",&th3);
     getfmt ("%8.5f",&th4);
     getfmt ("%8.5f",&th5);
     getfmt ("%8.5f",&th6);
     putfmt ("    th1       th2        th3         th4        th5");
     putfmt ("         th6\n");
     putfmt ("%9.3f %9.3f %9.3f %9.3f %9.3f %9.3f\n",
             th1,th2,th3,th4,th5,th6);
     if ( oat == 1)
     {
             th1 = th1 - 90;
             if ( th1 > 0 && th1 > MAX1 && th1 < MIN1 )
             {
                     putfmt (" limit on jt1 \n");
                     exit();
             }
             th2 = th2 + 90;
             if ( abs(th2) > MAX2 && abs(th2) < 180 )
             {
                     putfmt (" limit on jt2 \n");
                     exit();
             }
             th3 = th3 - 90;
             if ( th3 > 180 ) th3 = th3 - 360 ;
             if ( th3 < -180 ) th3 = th3 + 360 ;
             if ( abs(th3) > MAX3 && abs(th3) < 180 )
             {
                     putfmt (" limit on jt3 \n");
                     exit();
             }
             th6 = th6 + 90;
```

```
if ((th4 > MAX4 && th4 < 180) || (th4 < MIN4 && th4 > -180))
        {
                putfmt (" limit on jt4 \n");
                exit();
        }
        if ( abs(th5) > MAX5 && abs(th5) < 180 )
        {
                putfmt (" limit on jt5 \n");
                exit();
        }
        if ( th6 > MAX6 || th6 < MIN6 )
        {
                putfmt (" limit on jt6 \n");
                exit();
        }
}
putfmt (" th1d to th6d ; one per line \n");
getfmt ("%8.5f",&th1d);
getfmt ("%8.5f",&th2d);
getfmt ("%8.5f",&th3d);
getfmt ("%8.5f",&th4d);
getfmt ("%8.5f",&th5d);
getfmt ("%8.5f",&th6d);
putfmt ("   th1d      th2d       th3d       th4d       th5d");
putfmt ("         th6d\n");
putfmt ("%9.3f %9.3f %9.3f %9.3f %9.3f %9.3f\n",
        th1d,th2d,th3d,th4d,th5d,th6d);
th1d *= RAD;
th2d *= RAD;
th3d *= RAD;
th4d *= RAD;
th5d *= RAD;
th6d *= RAD;

putfmt (" th1dd to th6dd ; one per line \n");
getfmt ("%8.5f",&th1dd);
getfmt ("%8.5f",&th2dd);
getfmt ("%8.5f",&th3dd);
getfmt ("%8.5f",&th4dd);
getfmt ("%8.5f",&th5dd);
getfmt ("%8.5f",&th6dd);
putfmt ("  th1dd     th2dd      th3dd      th4dd      th5dd");
putfmt ("         th6dd\n");
putfmt ("%9.3f %9.3f %9.3f %9.3f %9.3f %9.3f\n",
        th1dd,th2dd,th3dd,th4dd,th5dd,th6dd);
```

```
th1dd *= RAD;
th2dd *= RAD;
th3dd *= RAD;
th4dd *= RAD;
th5dd *= RAD;
th6dd *= RAD;

jt_cart ();

putfmt (" k1 =%i k2 =%i k3 =%i\n",k1,k2,k3);
if ( euler == 1 )
{
        ephi = rho + 90;
        if (ephi > 180 ) ephi = ephi - 360;
        if (ephi < -180) ephi = ephi + 360;
        etheta = rtheta;
        eeps = reps - 90;
        if ( eeps > 180 ) eeps = eeps - 360 ;
        if ( eeps < -180) eeps = eeps + 360 ;
        putfmt("    rx       ry       rz      ephi    etheta");
        putfmt ("    eeps\n");
        putfmt ("%9.3f %9.3f %9.3f %9.3f %9.3f %9.3f\n",
                rx,ry,rz,ephi,etheta,eeps);
}
else if ( oat == 1 )
{
        o = rho + 180;
        a = rtheta - 90;
        t = reps -90;
        if ( o > 180 ) o = o - 360 ;
        if ( o < -180 ) o = o + 360 ;
        if ( a > 180 ) a = a - 360 ;
        if ( a < -180 ) a = a + 360 ;
        if ( t > 180 ) t = t - 360 ;
        if ( t < -180 ) t = t + 360 ;
        rz = rz - L1;
        putfmt("    rx       ry       rz        o         a");
        putfmt ("        t\n");
        putfmt ("%9.3f %9.3f %9.3f %9.3f %9.3f %9.3f\n",
                rx,ry,rz,o,a,t);
}
else
{
        putfmt("    rx       ry       rz      rho     rtheta");
        putfmt ("    reps\n");
```

```
                putfmt ("%9.3f %9.3f %9.3f %9.3f %9.3f %9.3f\n",
                        rx,ry,rz,rho,rtheta,reps);
        }

        wx *= DEG;
        wy *= DEG;
        wz *= DEG;

        putfmt("    vx        vy        vz        wx          wy");
        putfmt ("      wz\n");
        putfmt ("%9.3f %9.3f %9.3f %9.3f %9.3f %9.3f\n",
                vx,vy,vz,wx,wy,wz);

        wdx *= DEG;
        wdy *= DEG;
        wdz *= DEG;

        putfmt("    ax        ay        az        wdx          wdy");
        putfmt ("      wdz\n");
        putfmt ("%9.3f %9.3f %9.3f %9.3f %9.3f %9.3f\n",
                ax,ay,az,wdx,wdy,wdz);
}
else
{
        putfmt (" k1,k2,k3 ; one per line \n");
        getfmt ("%i",&k1);
        getfmt ("%i",&k2);
        getfmt ("%i",&k3);

        if (euler == 1)
        {
              putfmt ("rx,ry,rz,ephi,etheta,eeps;one per line(8.3)\n");
                getfmt ("%8.5f",&rx);
                getfmt ("%8.5f",&ry);
                getfmt ("%8.5f",&rz);
                getfmt ("%8.5f",&ephi);
                getfmt ("%8.5f",&etheta);
                getfmt ("%8.5f",&eeps);
                putfmt("    rx        ry        rz       ephi     etheta");
                putfmt ("     eeps\n");
                putfmt ("%9.3f %9.3f %9.3f %9.3f %9.3f %9.3f\n",
                        rx,ry,rz,ephi,etheta,eeps);
                rho = ephi - 90;
                rtheta = etheta ;
                reps = eeps + 90;
```

```
}
else if ( oat == 1 )
{
      putfmt ("rx,ry,rz,o,a,t; one per line(8.3)\n");
        getfmt ("%8.5f",&rx);
        getfmt ("%8.5f",&ry);
        getfmt ("%8.5f",&rz);
        getfmt ("%8.5f",&o);
        getfmt ("%8.5f",&a);
        getfmt ("%8.5f",&t);
        putfmt("    rx       ry       rz       o          a");
        putfmt ("       t\n");
        putfmt ("%9.3f %9.3f %9.3f %9.3f %9.3f %9.3f\n",
                rx,ry,rz,o,a,t);
        rz = rz + L1 ;
        rho = o - 180 ;
        rtheta = a + 90 ;
        reps = t + 90 ;
}
else
{
      putfmt ("rx,ry,rz,rho,rtheta,reps; one per line(8.3)\n");
        getfmt ("%8.5f",&rx);
        getfmt ("%8.5f",&ry);
        getfmt ("%8.5f",&rz);
        getfmt ("%8.5f",&rho);
        getfmt ("%8.5f",&rtheta);
        getfmt ("%8.5f",&reps);
        putfmt("    rx       ry       rz      rho     rtheta");
        putfmt ("    reps\n");
        putfmt ("%9.3f %9.3f %9.3f %9.3f %9.3f %9.3f\n",
                rx,ry,rz,rho,rtheta,reps);
}
putfmt ("vx,vy,vz,wx,wy,wz; one per line(8.3)\n");
getfmt ("%8.5f",&vx);
getfmt ("%8.5f",&vy);
getfmt ("%8.5f",&vz);
getfmt ("%8.5f",&wx);
getfmt ("%8.5f",&wy);
getfmt ("%8.5f",&wz);
putfmt("    vx       vy       vz       wx          wy");
putfmt ("      wz\n");
putfmt ("%9.3f %9.3f %9.3f %9.3f %9.3f %9.3f\n",
        vx,vy,vz,wx,wy,wz);
wx *= RAD ;
```

```
wy *= RAD ;
wz *= RAD ;

putfmt ("ax,ay,az,wdx,wdy,wdz; one per line(8.3) n");
getfmt ("%8.5f",&ax);
getfmt ("%8.5f",&ay);
getfmt ("%8.5f",&az);
getfmt ("%8.5f",&wdx);
getfmt ("%8.5f",&wdy);
getfmt ("%8.5f",&wdz);
putfmt("    ax       ay       az       wdx         wdy");
putfmt ("      wdz n");
putfmt ("%9.3f %9.3f %9.3f %9.3f %9.3f %9.3f n",
        ax,ay,az,wdx,wdy,wdz);
wdx *= RAD ;
wdy *= RAD ;
wdz *= RAD ;

cart_jt ();

switch (status)
{
        case -1:
                putfmt (" position unobtainable  n");
                break;
        case -2:
        putfmt (" th1d & th1dd unobtainable due to wb = 0  n");
                break;
        case -3:
putfmt ("th2d,th3d,th2dd & th3dd unobtainable due to s3p=0 n");
                break;
        case -4:
        putfmt (" th6d & th6dd unobtainable due to s5 = 0  n");
                break;
        case 1:
                putfmt (" limit on joint 1  n");
                break;
        case 2:
                putfmt (" limit on joint 2  n");
                break;
        case 3:
                putfmt (" limit on joint 3  n");
                break;
        case 4:
                putfmt (" limit on joint 4  n");
```

```
        break;
case 5:
        putfmt (" limit on joint 5  n");
        break;
case 6:
        putfmt (" limit on joint 6  n");
        break;
default:
        if ( oat == 1 )
        {
                th1 = th1 + 90 ;
                th2 = th2 - 90 ;
                th3 = th3 + 90 ;
                th6 = th6 - 90 ;
                if ( th1 > 180 ) th1 = th1 - 360 ;
                if ( th1 < -180 ) th1 = th1 + 360 ;
                if ( th2 > 180 ) th2 = th2 - 360 ;
                if ( th2 < -180 ) th2 = th2 + 360 ;
                if ( th3 > 180 ) th3 = th3 - 360 ;
                if ( th3 < -180 ) th3 = th3 + 360 ;
        }

putfmt ("    th1      th2      th3      th4      th5");
putfmt ("      th6\n");
putfmt ("%9.3f %9.3f %9.3f %9.3f %9.3f %9.3f\n",
        th1,th2,th3,th4,th5,th6);

th1d *= DEG;
th2d *= DEG;
th3d *= DEG;
th4d *= DEG;
th5d *= DEG;
th6d *= DEG;

putfmt ("   th1d     th2d     th3d     th4d     th5d");
putfmt ("      th6d\n");
putfmt ("%9.3f %9.3f %9.3f %9.3f %9.3f %9.3f\n",
        th1d,th2d,th3d,th4d,th5d,th6d);

th1dd *= DEG;
th2dd *= DEG;
th3dd *= DEG;
th4dd *= DEG;
th5dd *= DEG;
th6dd *= DEG;
```

```
                putfmt ("   th1dd      th2dd      th3dd      th4dd      ");
                putfmt ("       th5dd       th6dd\n");
                putfmt ("%9.3f %9.3f %9.3f %9.3f %9.3f %9.3f\n",
                        th1dd,th2dd,th3dd,th4dd,th5dd,th6dd);
            }
        }
    }
```

/* atan.c */

```
float atan2 ( m,n )          /* computes right quadrant for */
                             /*   the arc tangent function   */
float m,n;
{
        float ang;
        double arctan();

        if ( m == 0 && n == 0 )
        {
                putfmt (" INVALID PARAMETERS for atan2(m,n) !!\n");
                exit();
        }
        if ( n==0 ) return ( (m<0) ? -90.0 : 90.0 );
        if ( m==0 ) return ( (n<0) ? 180.0 :  0.0 );

        ang = DEG * (arctan((double) (abs(m) / abs(n)) ));

        if ( m>0 && n<0 ) return ( 180. - ang );
        if ( m<0 && n<0 ) return ( ang - 180. );
        if ( m<0 && n>0 ) return ( - ang );
        return ( ang);
}
```

```
/*   defin.h   */

#define  L1    672
#define  L2    432
#define  L3    433
#define  L4     56
#define  D     149.1
#define  AL     20.5
#define  ANG    2.72
#define  EPS    0.05
#define  RAD    0.0174533
#define  DEG    57.2957795
#define  ANGRAD 0.04742976
#define  MAX1   70
#define  MIN1   110
#define  MAX2   133
#define  MAX3   142
#define  MAX4   170
#define  MIN4   -110
#define  MAX5   100
#define  MAX6   356
#define  MIN6   -176
```

```
                        /*   kinempva.c   */

jt_cart ( )
                        /* transforms joint coord. to cartesian */
                        /* coord. (position, velocity & acceleration)*/
{
     extern float th1,th2,th3,th4,th5,th6;
     extern float rx,ry,rz,rho,rtheta,reps;
     extern float th1d,th2d,th3d,th4d,th5d,th6d;
     extern float th1dd,th2dd,th3dd,th4dd,th5dd,th6dd;
     extern float vx,vy,vz,wx,wy,wz;
     extern float ax,ay,az,wdx,wdy,wdz;
     extern int k1,k2,k3;

     double sin(),cos(),sqrt();
     double t1,t2,t3,t4,t5,t6,t23,t46;

     float  atan2(),c46,s46,rxw,ryw,rzw,i4x,i4y,i4z;
     float  c1,s1,c2,s2,c23,s23,c4,s4,c5,s5,crtheta,srtheta;
     float  a1,a2,num1,num2,den1,den2,rh,wa,wb,cr,sr,a,rr,num;
     float  w3x,w3y,w3z,w1x,w1y,w1z,vewx,vewy,vewz;
     float  v1wx,v1wy,v1wz,wd1x,wd1y,wd1z,wd4x,wd4y,wd4z,a1x,a1y,a1z;
     float  widx,widy,widz,we1x,we1y,we1z,awx,awy,awz;
     float  s5t6,c5t6,1c3,1s3,1c23,1s23,bx,by,fx,fy,t11,t22,th23d;

     t1 = th1 * RAD;
     t2 = th2 * RAD;
     t3 = th3 * RAD;
     t4 = th4 * RAD;
     t5 = th5 * RAD;
     t6 = th6 * RAD;

     t23 = t2 + t3 ;
     t46 = t4 + t6 ;

     c1 = cos ( t1 );
     s1 = sin ( t1 );
     c2 = cos ( t2 );
     s2 = sin ( t2 );
     c23 = cos ( t23 );
     s23 = sin ( t23 );
     c4 = cos ( t4 );
     s4 = sin ( t4 );
     c5 = cos ( t5 );
```

```
    s5 = sin ( t5 );


/* calculate rho, rtheta, reps */

    crtheta = c23 * c5 - s23 * s5 * c4;
    num2 = s5 * s4;
    if ( abs( crtheta ) != 1 )
    {
         a1 = s5 * c23;
         a2 = c5 * s23;
         num1 = s23 * s4;
         den1 = a1 + a2 * c4;
         den2 = a2 + a1 * c4;
         reps = th6 + atan2 ( num1 , den1 );
         rho = th1 + atan2 ( num2 , den2 );
         srtheta = sqrt ((double) (1 - crtheta * crtheta ));
         rtheta = atan2 ( srtheta, crtheta );
    }
    else
    {
         putfmt (" WARNING .. cos rtheta = 1 ..\n");
         c46 = cos ( t46 );
         s46 = sin ( t46 );
         if ( crtheta == 1) rtheta = 0;
         else rtheta = 180;
         srtheta = 0;
         reps = 0;
         num = ( 2 * s46 ) / ( c23 + crtheta );
         rho = th1 + atan2 ( num , c46 );
    }


    /* calculate angular velocity of end effector: wx,wy,wz */
    /* ang. vel. w.r.t. link 3 in link 3 coord: w3x,w3y,w3z */
    /* ang. vel. w.r.t. link 1 in link 1 coord: w1x,w1y,w1z */

    s5t6 = s5 * th6d ;
    c5t6 = c5 * th6d ;
    w3x = -c4 * th5d - num2 * th6d;
    w3y = -s4 * th5d + c4 * s5t6;
    w3z = th4d + c5t6;

    th23d = th2d + th3d ;
    w1x = w3x - th23d ;
```

```
w1y = c23 * w3y + s23 * w3z;
w1z = c23 * w3z - s23 * w3y;

wx = c1 * w1x - s1 * w1y;
wy = c1 * w1y + s1 * w1x;
wz = w1z + th1d;


/* calculate angular accel. of end effector: wdx,wdy,wdz */
/* ang. accel. w.r.t. wrist in link 4 coord.: wd4x,wd4y,wd4z */
/* ang. accel. w.r.t. wrist in link 1 coord.: wd1x,wd1y,wd1z */

wd4x = - th5dd - th4d * s5t6 ;
wd4y = s5 * th6dd + th5d * ( - th4d + c5t6 );
wd4z = th4dd + c5 * th6dd - s5t6 * th5d;

by = s4 * wd4x + c4 * wd4y ;
wd1x = c4 * wd4x - s4 * wd4y ;
wd1y = c23 * by + s23 * wd4z;
wd1z = -s23 * by + c23 * wd4z ;

fx = wd1x - th2dd - th3dd - th1d * w1y ;
fy = wd1y + th1d * ( th23d + w1x ) + th23d * wz ;

wdx = c1 * fx - s1 * fy ;
wdy = s1 * fx + c1 * fy ;
wdz = wd1z + th1dd - th23d * w1y ;


/* calculate rx,ry,rz */

a = L4 * srtheta;
rr = rho * RAD;
sr = sin ((double) (rr));
cr = cos ((double) (rr));
t23 -= ANGRAD ;
c23 = cos ( t23 );
s23 = sin ( t23 );

lc23 = L3 * c23;
ls23 = L3 * s23 ;
wa = c2 * L2 + lc23 ;
wb = s2 * L2 + ls23 ;

rxw = -s1 * wb - D * c1 ;
```

```
      ryw = c1 * wb - D * s1 ;
      rzw = wa + L1 ;

      i4x = - a * sr ;
      i4y = a * cr ;
      i4z = L4 * crtheta ;

      rx = rxw + i4x;
      ry = ryw + i4y;
      rz = rzw + i4z;


   /* calculate linear vel. of end effector: vx, vy, vz */
   /* linear vel. of end eff. w.r.t. wrist ( we x i4 ): vewx,vewy,vewz */
   /* linear vel. of wrist w.r.t. base (link 1 coord): v1wx,v1wy,v1wz */

      ls3 = ls23 * th3d ;
      lc3 = lc23 * th3d ;

      vewx = wy * i4z - wz * i4y;
      vewy = wz * i4x - wx * i4z;
      vewz = wx * i4y - wy * i4x;

      bx = wa * th2d + lc3 ;
      v1wx = - wb * th1d;
      v1wy = - D * th1d + bx ;
      v1wz = - wb * th2d - ls3 ;

      vx = vewx + c1 * v1wx - s1 * v1wy;
      vy = vewy + s1 * v1wx + c1 * v1wy;
      vz = vewz + v1wz;


      /* calculate linear accel. of end effector: ax, ay, az      */
      /* linear accel. of wrist w.r.t. link 1 coord.: a1x,a1y,a1z */
      /* linear accel. of wrist: awx, awy, awz                    */
      /* cross product ang. accel. and i4 : widx,widy,widz        */
      /* cross product we x ( we x i4 ): weix,weiy,weiz           */

      t11 = th1d * th1d;
      t22 = th2d * th2d ;
      th23d += th2d ;
      a1x = - wb * th1dd + D * t11 - 2 * th1d * bx ;
      a1y = -D*th1dd + wa*th2dd + lc23*th3dd - (t11+t22)*wb - ls3*th23d;
      a1z = - wb * th2dd - ls23 * th3dd - t22 * wa - lc3 * th23d;
```

```
    awx = c1 * alx - s1 * aly ;
    awy = s1 * alx + c1 * aly ;
    awz = alz ;

    widx = wdy * i4z - wdz * i4y ;
    widy = wdz * i4x - wdx * i4z ;
    widz = wdx * i4y - wdy * i4x ;

    weix = wy * vewz - wz * vewy ;
    weiy = wz * vewx - wx * vewz ;
    weiz = wx * vewy - wy * vewx ;

    ax = awx + weix + widx ;
    ay = awy + weiy + widy ;
    az = awz + weiz + widz ;



    /* arm configuration */
    if ( wb < 0 ) k1 = -1;
    if ( k1*th3 < 0 ) k2 = -1;
    if ( th5 < 0 ) k3 = -1;
}



cart_jt ()                  /* transforms cartesian  coord. to */
                            /* joint coord.                    */

{
    extern float th1,th2,th3,th4,th5,th6;
    extern float rx,ry,rz,rho,rtheta,reps;
    extern float th1d,th2d,th3d,th4d,th5d,th6d;
    extern float th1dd,th2dd,th3dd,th4dd,th5dd,th6dd;
    extern float vx,vy,vz,wx,wy,wz;
    extern float ax,ay,az,wdx,wdy,wdz;
    extern int status,k1,k2,k3;

    double cos(),sin(),sqrt(),t23,t23p,t1,t4;

    float atan2(),i4x,i4y,i4z;
    float w1[3],wa,n2,k,k12;
    float num1,den1,c3p,s3p,wb,a2,b2;
```

```
float c23,s23,cr1,sr1,c5,s5,c4,s4,s4b;
float ctheta,stheta,crho,srho,cbeta,sbeta,beta;
float rth,rh,a,len,s1,c1,c23p,s23p,b;
float vwx,vwy,vwz,v1wx,v1wy,v1wz,w1x,w1y,w1z,w3x,w3y,w3z;
float wix,wiy,wiz,weix,weiy,weiz,widx,widy,widz,fx,fy,fz;
float awx,awy,awz,a1x,a1y,a1z,wd4x,wd4y,wd4z,wd1x,wd1z,wd1y;
float t11,t22,bx,123,1s3,1c3,k13,th23d,c5t6,s5t6;

rth = rtheta * RAD;
rh = rho * RAD;
ctheta = cos (rth);
stheta = sin (rth);
crho = cos (rh);
srho = sin (rh);

a = L4 * stheta;
i4x = - a * srho ;
i4y = a * crho ;
i4z = L4 * ctheta ;

w1[0] = rx - i4x;
w1[1] = ry - i4y;
w1[2] = rz - i4z;


/* calculate th1, th2, th3 */

len = (double) ( w1[0] * w1[0] + w1[1] * w1[1] - D*D );
if ( len < 0 )   return ( status = -1 );
len = sqrt ( len );
wb = k1 * len ;
if ( wb == 0 ) return ( status = -2 );

if ( k1 > 0 )   th1 = atan2 (-w1[0],w1[1]) - atan2 (D,len);
else  th1 = atan2 (w1[0],-w1[1]) + atan2 (D,len);
if ( th1 > 180 ) th1 = th1 - 360 ;

/* limit check on th1 */
if ( th1 > 0 && th1 > MAX1 && th1 < MIN1 ) return (status = 1);

t1 = th1 * RAD ;
c1 = cos ( t1 ) ;
s1 = sin ( t1 ) ;
wa = w1[2] - L1 ;
n2 = len * len + wa * wa;
```

```
c3p = ( n2 - L2 * L2 - L3 * L3 ) / ( 2 * L2 * L3 );
if ( abs(c3p) > 1.001 ) return (status= -1);
                          /* causes main to stop the arm */
                          /*and to give following message*/
                          /* " position unobtainable "   */

if ( abs(c3p) <= 1.001 && abs(c3p) > 1 ) c3p = c3p /abs(c3p);
if (abs(c3p) == 1) {
            putfmt ("WARNING.. maximum reach -- jt3 \n");
            return ( status = -3 );
    }
s3p = sqrt ((double) ( 1 - c3p * c3p ));
k12 = k1 * k2 ;
th3 = k12 * atan2 (s3p,c3p) + ANG ;

/* limit check for th3 */
if ( abs(th3) > MAX3 && abs(th3) < 180 ) return ( status = 3);

a2 = L3 * s3p;
b2 = L2 + L3 * c3p;
th2 = k1 * ( atan2( len , wa) - k2 * atan2 (a2,b2)) ;

/* limit check for th2 */
if ( abs(th2) > MAX2 && abs(th2) < 180 ) return ( status = 2 );


/* calculate th1d, th2d, th3d */
/* wrist linear velocity w.r.t. base in base coords: vwx,vwy,vwz */
/* wrist linear velocity in link 1 coord. : v1wx,v1wy,v1wz */

t23 = RAD * ( th2 + th3 );
c23 = cos (t23);
s23 = sin (t23);
t23p = t23 - ANGRAD;
c23p = cos (t23p);
s23p = sin (t23p);

wix = wy * i4z - wz * i4y ;
wiy = wz * i4x - wx * i4z ;
wiz = wx * i4y - wy * i4x ;

vwx = vx - wix ;
vwy = vy - wiy ;
vwz = vz - wiz ;
```

```
vlwx = c1 * vwx + s1 * vwy ;
vlwy = c1 * vwy - s1 * vwx ;
vlwz = vwz ;

th1d = - vlwx / wb ;
b = vlwy + th1d * D ;
k = k12 * s3p * L2 ;          /* multiply by k12 to give s3p its
                                 correct sign */
k13 = k * L3 ;
th2d = ( s23p * b + c23p * vlwz ) / k ;
th3d = ( wb * b + wa * vlwz ) / ( - k13 );


/* calculate joint accelerations th1dd, th2dd, th3dd          */
/* linear accel of wrist w.r.t. base coord.: awx,awy,awz      */
/* cross product we x ( we x i4 ) : weix,weiy,weiz            */
/* cross product wed x i4 : widx,widy,widz                    */

widx = wdy * i4z - wdz * i4y ;
widy = wdz * i4x - wdx * i4z ;
widz = wdx * i4y - wdy * i4x ;

weix = wy * wiz - wz * wiy ;
weiy = wz * wix - wx * wiz ;
weiz = wx * wiy - wy * wix ;

awx = ax - weix - widx ;
awy = ay - weiy - widy ;
awz = az - weiz - widz ;

alx = c1 * awx + s1 * awy ;
aly = - s1 * awx + c1 * awy ;
alz = awz ;

t11 = th1d * th1d;
t22 = th2d * th2d ;
l23 = L3 * th3d ;
ls3 = l23 * s23p ;
lc3 = l23 * c23p ;
th23d = 2 * th2d + th3d ;

fx = alx - D * t11 + 2 * th1d * ( th2d * wa + lc3 );
fy = aly + ( t11 + t22 ) * wb + ls3 * th23d ;
fz = alz + t22 * wa + lc3 * th23d;
```

```
th1dd = - fx / wb ;
th2dd = ( s23p * ( -D * fx / wb + fy ) + c23p * fz )/ k;
th3dd = ( D * fx - wb * fy - wa * fz ) / k13 ;


/* calculate th4, th5, th6 */

cr1 = cos ( rh - t1 );
sr1 = sin ( rh - t1 );
c5 = c23 * ctheta + s23 * stheta * cr1;
s5 = sqrt ((double) ( 1 - c5 * c5 ));
th5 = atan2 ( s5, c5);
if ( s5   0.001) {
     sbeta = s23 * sr1;
     cbeta = stheta * c23 - ctheta * s23 *cr1;
     beta = atan2 (sbeta, cbeta );
     th6 = reps - beta ;

     if ( s5   EPS )
     {
             s4 = stheta * sr1;
             c4 = c23 * stheta *cr1 - s23 * ctheta ;
             th4 = atan2 ( s4, c4 );
     }
     else
     {
             s4b = 0.5 * sr1 * ( c23 + ctheta );
             th4 = atan2 ( s4b, cr1) + beta ;
     }
}
else                              /* straight wrist */
{
     putfmt ("WARNING .. straight wrist  n");
     return ( status = -4 );
/*   beta = 0;
     th6 = reps ;
     s4b = 0.5 * sr1 * ( c23 + ctheta );
     th4 = atan2 ( s4b, cr1) + beta ;        */
}

/* limit check for th4, th5, th6 */
if ( abs(th5)   MAX5 && abs(th5)   180 ) return ( status = 5 );
if (((th4   MAX4 && th4   180 ) || (th4   MIN4 && th4   -180 ))
   || ( th6 > MAX6 || th6 < MIN6 ) || ( k3 == -1 ))
```

```
{
     th5 = - th5;
     th4 = th4 + 180;
     if ( th4 > 180 ) th4 = th4 - 360;
     if ( th4 < -180 ) th4 = th4 + 360;
     th6 = th6 + 180;
     if ( th6 > 360 ) th6 = th6 - 360;
}
if (( th4 > MAX4 && th4 < 180 ) || ( th4 < MIN4 && th4 > -180 ))
     return ( status = 4 );
if ( th6 > MAX6 || th6 < MIN6 ) return ( status = 6 );

/* calculate th4d, th5d, th6d */
/* ang vel of end eff. w.r.t. link3 in link1 coord: w1x,w1y,w1z */
/* ang vel of end eff. w.r.t. link3 in link3 coord: w3x,w3y,w3z */

th23d = th2d ;

w1x = c1 * wx + s1 * wy ;
w1y = c1 * wy - s1 * wx ;
w1z = wz - th1d ;

w3x = w1x + th23d ;
w3y = c23 * w1y - s23 * w1z ;
w3z = c23 * w1z + s23 * w1y ;

t4 = th4 * RAD ;
c4 = cos ( t4 );
s4 = sin ( t4 );

if ( th5 < 0 )  s5 = - s5 ;
th5d = - c4 * w3x - s4 * w3y ;
th6d = ( c4 * w3y - s4 * w3x ) / s5 ;
s5t6 = s5 * th6d ;
c5t6 = c5 * th6d ;
th4d = w3z - c5t6 ;


/* calculate joint accel. th4dd, th5dd, th6dd                */
/* ang. accel. w.r.t. wrist in link 1 coord.:wd1x,wd1y,wd1z  */
/* ang. accel. w.r.t. wrist in link 4 coord.:wd4x,wd4y,wd4z  */

wd1x = c1 * wdx + s1 * wdy + th2dd + th3dd + th1d * w1y ;
wd1y = c1 * wdy - s1 * wdx - ( th1d + wz ) * th23d - th1d * w1x ;
wd1z = wdz - th1dd + th23d * w1y ;
```

```
        bx = c23 * wd1y - s23 * wd1z ;
        wd4x = wd1x * c4 + bx * s4 ;
        wd4y = - wd1x * s4 + bx * c4 ;
        wd4z = c23 * wd1z + s23 * wd1y ;

        th5dd = - wd4x - th4d * s5t6 ;
        th6dd = ( wd4y + th5d * ( th4d - c5t6 )) / s5 ;
        th4dd = wd4z + th5d * s5t6 - c5 * th6dd ;


}
```